Azure K8s 생성

1. Azure 로그인
   1. <http://portal.azure.com> : jmjeon0827@outlook.com/t1!

* 구독 클릭 해서 구독정보 확인 하기

1. 리소스 그룹
   1. 리소스 그룹 검색
   2. 리소스 그룹 만들기
   3. 리소스그룹 : skteam33-rsrcgrp
   4. 영역 : 한국 중부
2. Kubernetes 서비스 생성
   1. kubernetes 서비스 검색
   2. kubernetes 클러스터 추가
   3. 리소스그룹 : skteam33-rsrcgrp
   4. kubernetes 클러스터 이름 : skteam33-aks
   5. 크기조정 방법 : 수동
   6. 노드 개수 : 1
   7. [만들기] 클릭 – 배포과정 몇 분 소용 됨
3. 컨테이너 레지스트리 검색
   1. 리소스그룹 : skteam33-rsrcgrp
   2. 레지스트리 이름 : skteam33
   3. 위치 : 한국 중부
   4. 이미지레지스트리 경로 : skteam33.azurecr.io
   5. [만들기]
4. AKS 클러스터 접속 및 ACR 연결
   1. Azure portla의 Cloud shell를 유료로 사용하거나 로컬 PC에서 CMD창 또는 PowerShell 실행
   2. Azure 로그인 : az login
   3. AKS 접속 : az aks get-credentials --resource-group skteam33-rsrcgrp --name skteam33-aks
   4. 서비스 확인 : kubectl get all
   5. ACR 로그인 : az acr login --name skteam33
   6. 클러스터와 ACR 연결(한번만 실행 하면 됨) : az aks update -n skteam33-aks -g skteam33-rsrcgrp --attach-acr skteam33
5. Maven 빌드 및 ACR Push
   1. Order

cd C:\Lv2Assessment\Source\SampleTest3\Order

mvn package

az acr build --registry skteam33 --image skteam33.azurecr.io/order:v1 .

* 1. Pay

cd C:\Lv2Assessment\Source\SampleTest3\Pay

mvn package

az acr build --registry skteam33 --image skteam33.azurecr.io/pay:v1 .

* 1. Delivery

cd C:\Lv2Assessment\Source\SampleTest3\Delivery

mvn package

az acr build --registry skteam33 --image skteam33.azurecr.io/delivery:v1 .

* 1. MyPage

cd C:\Lv2Assessment\Source\SampleTest3\MyPage

mvn package

az acr build --registry skteam33 --image skteam33.azurecr.io/mypage:v1 .

* 1. GateWay

cd C:\Lv2Assessment\Source\SampleTest3\gateway

mvn package

az acr build --registry skteam33 --image skteam33.azurecr.io/gateway:v1 .

\*\*\* 참고

az acr build --registry skteam33 --image skteam33.azurecr.io/oder:v1 .

는 아래와 같은 기능을 함

-- 도커 빌드

docker build -t skteam33.azurecr.io/order:v1 .

-- 도커 푸시

docker push skteam33.azurecr.io/order:v1

1. Pod 및 Service 배포
   1. Order

cd C:\Lv2Assessment\Source\SampleTest3\Order\kubernetes

kubectl apply -f deployment.yml

또는kubectl create deploy order --image=skteam33.azurecr.io/order:v1

kubectl apply -f service.yaml

* 1. Pay

cd C:\Lv2Assessment\Source\SampleTest3\Pay\kubernetes

kubectl apply -f deployment.yml

kubectl apply -f service.yaml

* 1. Delivery

cd C:\Lv2Assessment\Source\SampleTest3\Delivery\kubernetes

kubectl apply -f deployment.yml

kubectl apply -f service.yaml

* 1. MyPage

cd C:\Lv2Assessment\Source\SampleTest3\MyPage\kubernetes

kubectl apply -f deployment.yml

kubectl apply -f service.yaml

* 1. GateWay

cd C:\Lv2Assessment\Source\SampleTest3\gateway\kubernetes

kubectl apply -f deployment.yml

kubectl apply -f service.yaml

1. Kafka 설치
   1. chocolatey 설치(<https://evandde.github.io/chocolatey> 참고)

* cmd.exe를 관리자 권한으로 실행합니다.
* 다음 명령줄을 실행합니다.

@"%SystemRoot%\System32\WindowsPowerShell\v1.0\powershell.exe" -NoProfile -InputFormat None -ExecutionPolicy Bypass -Command " [System.Net.ServicePointManager]::SecurityProtocol = 3072; iex ((New-Object System.Net.WebClient).DownloadString('https://chocolatey.org/install.ps1'))" && SET "PATH=%PATH%;%ALLUSERSPROFILE%\chocolatey\bin"

* 1. Helm 설치(<https://helm.sh/ko/docs/intro/install/> - Windows)

cmd.exe에서 아래 명령어 실행 .

choco install kubernetes-helm

* 1. Azure Only????????????????????????????????

kubectl patch storageclass managed -p '{"metadata": {"annotations":{"storageclass.kubernetes.io/is-default-class":"true"}}}'

* 1. Helm 에게 권한을 부여하고 초기화

kubectl --namespace kube-system create sa tiller

kubectl create clusterrolebinding tiller --clusterrole cluster-admin --serviceaccount=kube-system:tiller

* 1. Kafka 설치 및 실행

helm repo add incubator https://charts.helm.sh/incubator

helm repo update

kubectl create ns kafka

helm install my-kafka --namespace kafka incubator/kafka

kubectl get po -n kafka -o wide

* 1. Kafka 실행 여부

kubectl -n kafka exec -it my-kafka-0 -- /bin/sh

ps –ef | grep kafka

* 1. Topic 생성

kubectl -n kafka exec my-kafka-0 -- /usr/bin/kafka-topics --zookeeper my-kafka-zookeeper:2181 --topic store --create --partitions 1 --replication-factor 1

* 1. Topic 확인

kubectl -n kafka exec my-kafka-0 -- /usr/bin/kafka-topics --zookeeper my-kafka-zookeeper:2181 --list

* 1. 이벤트 발행하기

kubectl -n kafka exec -ti my-kafka-0 -- /usr/bin/kafka-console-producer --broker-list my-kafka:9092 --topic store

* 1. 이벤트 수신하기

kubectl -n kafka exec -ti my-kafka-0 -- /usr/bin/kafka-console-consumer --bootstrap-server my-kafka:9092 --topic store --from-beginning

1. ConfigMap 생성 및 정보 확인

kubectl create configmap systemmode --from-literal=sysmode=PRODUCT

kubectl get configmap systemmode -o yaml

1. Httpie Pod 생성

cd C:\Lv2Assessment\Source\SampleTest3\Util\httpie\kubernetes

kubectl apply -f deployment.yml

kubectl exec -it httpie bin/bash

1. Siege Pod 생성

cd C:\Lv2Assessment\Source\SampleTest3\Util\siege\kubernetes

kubectl apply -f deployment.yml

kubectl exec -it siege bin/bash

1. 테스트
   1. 조회

http http://20.200.207.114:8080/orders

http http://20.200.207.114:8080/pays

http http://20.200.207.114:8080/deliveries

http http://20.200.207.114:8080/myPages

* 1. order 등록

http POST http://20.200.207.114:8080/orders customerId="customer1" customerName="홍길동1" productId="prd1" productName="prdName1" qty=10 amount=1000 orderStatus="A" address="ABC"

* 1. kafka 이벤트 수신

kubectl -n kafka exec -ti my-kafka-0 -- /usr/bin/kafka-console-consumer --bootstrap-server my-kafka:9092 --topic store --from-beginning

* 1. Siege

kubectl exec -it pod/siege -c siege -- /bin/bash

siege -c100 -t30S -v --content-type "application/json" "http://20.200.207.114:8080/orders"

'

1. 명령어 정리
   1. get

kubectl get pod –o wide

kubectl get deploy –o wide

kubectl get deploy –w

kubectl get service –o wide

* 1. Log

kubectl describe pod order

kubectl logs –f pod-asdfsadfsd-12312

kubectl exec –it pod-sadfsdfaa-23343-- /bin/sh

* 1. Delete

kubectl delete service,deploy Order

kubectl delete service,deploy pay

kubectl delete service,deploy delivery

kubectl delete service,deploy mypage

kubectl delete service,deploy gateway

kubectl delete deploy,service –all

* 1. 이미지 변경

kubectl set image deploy order order=skteam33.azurecr.io/order:v1

* 1. Autoscale

kubectl autoscale deploy order --min=1 --max=10 --cpu-percent=15